Evolving Feature Selection

Huan Liu, *Arizona State University*

Data preprocessing is an indispensable step in effective data analysis. It prepares data for data mining and machine learning, which aim to turn data into business intelligence or knowledge. Feature selection is a preprocessing technique commonly used on high-dimensional data. Feature selection studies how to select a subset or list of attributes or variables that are used to construct models describing data. Its purposes include reducing dimensionality, removing irrelevant and redundant features, reducing the amount of data needed for learning, improving algorithms' predictive accuracy, and increasing the constructed models' comprehensibility.

Feature selection is different from feature extraction (for example, principal component analysis, singular-value decomposition, manifold learning, and factor analysis), which creates new (ex-tracted) features that are combinations of the original features. Feature-selection methods are particularly welcome in interdisciplinary collaborations because the selected features retain the original meanings domain experts are familiar with. The rapid developments in computer science and engineering allow for data collection at an unprecedented speed and present new challenges to feature selection.

Wide data sets, which have a huge number of features but relatively few instances, introduce a novel challenge to feature selection. This type of data supports a vast number of models. Examples of such data are microarrays, transaction logs, and Web data. Unlabeled data presents another challenge; the lack of class labels compounds the already difficult problem of feature selection. The integration of domain knowledge in feature selection tenders a perpendicular challenge. Feature-selection methods attempt to explore data's intrinsic properties by employing statistics or information theory. As in microarray data analysis, however, statistical significance might not directly translate to biological relevance. It's imperative to integrate both in selecting features to generate results meaningful to domain experts.

These six articles address emerging issues that concern evolving feature selection. Edward Dougherty addresses the daunting small-sample problem, while Jennifer Dy introduces ways to select features for unlabeled data. Kari Torkkola and Eugene Tuv advocate achieving stability of feature selection with ensemble methods. Hanchuan Peng, Chris Ding, and Fuhui Long apply dual criteria (minimum redundancy and maximum relevance) to selecting features for microarray data. Michael Berens and his colleagues show how to foster biological relevance in feature selection. Finally, George Forman reviews feature selection's status and points out the need for further research.

—Huan Liu

Feature-Selection Overfitting with Small-Sample Classifier Design

Edward R. Dougherty, *Texas A&M University*

High-throughput technologies facilitate the measurement of vast numbers of biological variables, thereby providing enormous amounts of multivariate data with which to model biological processes.[1] In translational genomics, phenotype classification via gene expression promises highly discriminatory molecular-based diagnosis, and regulatory-network modeling offers the potential to develop therapeutic strategies based on genomic decision making using classical engineering disciplines such as control theory.[2] Yet one must recognize the obstacles inherent in dealing with extremely large numbers of interacting variables in a nonlinear, stochastic, and redundant system that reacts aggressively to any attempt to probe it—a living system. In particular, large data sets may have the perverse effect of limiting the amount of scientific information that can be extracted, because the ability to build models with scientific validity is negatively impacted by an increasing ratio between the number of variables and the sample size. Our specific interest is in how this dimensionality problem creates the need for feature selection while making feature-selection algorithms less reliable with small samples.

Two well-appreciated issues tend to confound feature selection: *redundancy* and *multivariate prediction*. Both of these can be illustrated by taking a naïve approach to feature selection by considering all features in isolation, ranking them on the basis of their individual predictive capabilities, selecting some features with the highest individual performances, and then applying a standard classification rule to these features, the reasoning being that these are the best predictors of the class. Redundancy arises because the top-performing features might be strongly related—say, by the fact that they share a similar regulatory pathway—and using more than one or two of them may provide little added benefit. The issue of multivariate prediction arises because top-performing single features may not be significantly more beneficial when used in combination with other features, whereas features that perform poorly when used alone may provide outstanding classifi-

1541-1672/05/$20.00 © 2005 IEEE
Published by the IEEE Computer Society

cation when used in combination. This situation can be dramatic in highly complex regulatory systems. Another impediment to feature selection concerns estimation. With small samples, the choice of error estimator can make a greater difference than the manner of feature selection.[3]

Overfitting

While redundancy, multivariate prediction, and error estimation can severely impact feature selection, my commentary here is aimed at the role of feature selection in *overfitting* the data and how this is exacerbated by high dimensionality and small samples.

A classification rule chooses a classifier from a family G of classifiers on the basis of the data. A classifier is optimal in G if its error, ε_G, is minimal among all classifiers in G. Since a designed classifier depends on the particular sample, it is random relative to random sampling. We would like the expected error, $\varepsilon_{G,n}$, of the designed classifier, n denoting sample size, to be close to ε_G.

If G and H are families of classifiers such that $G \subset H$, then $\varepsilon_H \leq \varepsilon_G$. However, the error relation need not hold for designed classifiers, where it may be that $\varepsilon_{H,n} > \varepsilon_{G,n}$. This is known as overfitting: the designed classifier partitions the feature space well relative to the sample data but not relative to the full distribution. Overfitting is ubiquitous for small samples. To mitigate overfitting, one can choose from smaller classifier families whose classifiers partition the feature space more coarsely. Using G instead of H, where $G \subset H$, reduces the *design cost*, $\varepsilon_{G,n} - \varepsilon_G$, relative to $\varepsilon_{H,n} - \varepsilon_H$ at the expense of introducing a *constraint cost*, $\varepsilon_G - \varepsilon_H$.

Consider a collection of classifier families, $G_1 \subset G_2 \subset G_3 \subset \ldots$, for a fixed sample size n. A typical situation might be that, while the smaller families extensively reduce design cost, their constraint is excessive. Thus, we might expect the expected errors of the designed classifiers to fall as we utilize increasingly large families but then to begin to increase when the design cost grows too much. Applying this reasoning to a sequence, $x_1, x_2, \ldots, x_d, \ldots$, of features, we might expect at first a decrease in expected error as d increases and then subsequently an increase in error for increasing d. While this description is idealized and the situation can be more complex, it describes the *peaking phenomenon*. In this scenario, one would be interested in the optimal number of features.[4]

In practice, the features are not ordered, and the best feature set must be found from among all possible feature subsets. We confront a fundamental limiting principle: In the absence of countervailing distribution knowledge, to select a subset of k features from a set of features and be assured that it provides minimum error among all optimal classifiers for subsets of size k, all k-element subsets must be checked.[5] Thus, we are challenged to find suboptimal feature-selection algorithms.

When used, a feature-selection algorithm is part of the classification rule. This is why feature selection must be included when using cross-validation error estimation. Feature selection yields classifier constraint, not a reduction in the dimensionality of the feature space relative to design. For instance, if there are d features available for linear discriminant analysis (LDA), when used directly, then the classifier family consists of all hyperplanes in d-dimensional space. But, if a feature-selection algorithm reduces the number of variables to $m < d$ prior to application of LDA, then the classifier family consists of all hyperplanes in d-dimensional space confined to m-dimensional subspaces. The dimensionality of the classification rule has not been reduced, but the new classification rule (feature selection plus LDA) is constrained. The issue is whether it is sufficiently constrained. Given 20,000 gene-expression levels as features, the new rule has significant potential for overfitting.

When used, a feature-selection algorithm is part of the classification rule. This is why feature selection must be included when using cross-validation error estimation.

An illustration

For illustration, consider a d-dimensional model where the class conditional densities for 0 and 1 are uniformly distributed over the regions

$$D_0 = [0, a_1/2] \times [0, a_2] \times [0, a_3] \times \ldots \times [0, a_d]$$
$$D_1 = [a_1/2, 1] \times [0, a_2] \times [0, a_3] \times \ldots \times [0, a_d]$$

respectively, $a_1, a_2, \ldots, a_d > 0$. This model is useful to illuminate the difficulty of small-sample feature selection for several reasons. First, for the first feature there is a perfect classifier (0 error) consisting of the single-point decision boundary, $x_1 = a_1/2$; for every other feature x_k, every classifier consisting of a finite number of splits of the interval $[0, a_k]$ has error 0.5; and so long as x_1 is not included, every feature set composed of any number of variables has error 0.5. Thus, in some sense, this corresponds to the easiest possible feature-selection problem. Second, it is not complicated by redundancy because all features are independent. Third, it is not complicated by multivariate prediction because optimal feature selection involves a single feature, x_1. Finally, the problem is not necessarily mitigated by the common practice of commencing feature selection by throwing out those with low variance. As seen by the error formulas below, the variances of the features play no role, and one might eliminate the good feature if a_1 is small in comparison to $a_2, a_3, \ldots, a_d$.

I now demonstrate that high dimensionality and low sample size can make finding the 0-error feature difficult even though all other features have error 0.5. We consider three single-feature classification rules of increasing complexity: a single split, up to two splits, and up to three splits of the interval. The probabilities of a random sample of size $2n$, equally split between the two classes, being perfectly separated by a single value, at most two values, and at most three values of x_k are

$$p_{1,n} = \frac{2(n!)^2}{(2n)!}$$

$$p_{2,n} = \frac{2n(n!)^2}{(2n)!}$$

$$p_{3,n} = \frac{2\left(n^2 - n + 1\right)(n!)^2}{(2n)!}$$

respectively. Since the feature distribution is uniform, all features are independent. Therefore, the separability or lack of separability of the data by features $x_2, x_3, \ldots, x_d$ constitutes a binomial distribution with $d - 1$

Table 1. Expected number of separating feature sets using one split.

$2n\backslash(d-1)$	1,000	5,000	20,000
10	8	40	159
12	2	11	43
14	1	3	12
16		1	3
18			1
20			

Table 2. Expected number of separating feature sets using at most two splits.

$2n\backslash(d-1)$	1,000	5,000	20,000
10	40	198	793
12	13	65	260
14	4	20	81
16	1	12	50
18		2	7
20		1	2

Table 3. Expected number of separating feature sets using at most three splits.

$2n\backslash(d-1)$	1,000	5,000	20,000
10	167	833	3,333
12	67	336	1,342
14	25	125	501
16	9	44	177
18	3	15	60
20	1	5	20

trials. Thus, the expected number, $N_{k,d,n}$, of separating features is $E[N_{k,d,n}] = (d-1)p_{k,n}$ for $k = 1$, 2, and 3 splits.

The danger of obtaining a poor feature set with high-dimensional data sets and small samples is seen in tables 1 through 3, which give $E[N_{k,d,n}]$ for large values of $d - 1$ and small sample sizes. And this is for a situation in which every poorly selected feature has error 0.5! More tables giving $E[N_{k,d,n}]$ are provided at www.ee.tamu.edu/~edward/feature_overfitting.

In conclusion, note that in high-dimension, small-sample settings, a key difficulty is the masking of good feature sets by bad ones. The result can be false-negative reasoning where one wrongly concludes that no good feature sets exist simply because they cannot be found. Owing to its importance for high-throughput technologies, feature selection is receiving much attention with many schemes being proposed. It seems incumbent on those proposing algorithms that limitations be investigated at the outset to see under what conditions one can reasonably expect satisfactory results.

References

1. J. Chen et al., "Grand Challenges for Multimodal Bio-Medical Systems," *IEEE Circuits and Systems*, vol. 5, no. 2, 2005, pp. 46–52.
2. E.R. Dougherty and A. Datta, "Genomic Signal Processing: Diagnosis and Therapy," *IEEE Signal Processing*, vol. 22, no. 1, 2005, pp. 107–112.
3. C. Sima et al., "Impact of Error Estimation on Feature-Selection Algorithms," *Pattern Recognition*, vol. 38, no. 12, 2005, pp. 2472–2482.
4. J. Hua et al., "Optimal Number of Features as a Function of Sample Size for Various Classification Rules," *Bioinformatics*, vol. 21, no. 8, 2005, pp. 1509–1515.
5. T. Cover and J. Van Campenhout, "On the Possible Orderings in the Measurement Selection Problem," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 7, no. 9, 1977, pp. 657–661.

Feature Selection for Unlabeled Data

Jennifer G. Dy, *Northeastern University*

Technological advances such as the Internet, hyperspectral imagery, microarrays, and digital-storage-capacity increases have contributed to the existence of large volumes of data. One way to extract information and knowledge from data is through clustering or unsupervised learning.

Creating effective algorithms for unsupervised learning, or learning from unlabeled data, is important because large amounts of data preclude humans from manually labeling the categories of each instance. In addition, human labeling is expensive and subjective. So, much existing data is unlabeled.

Unsupervised learning, or cluster analysis, aims to group similar objects. A metric or a probability model typically defines *similarity*. These measures depend highly on the features representing the data. Many clustering algorithms assume that domain experts have determined relevant features. But not all features are important; some might be redundant or irrelevant. And the presence of many irrelevant features can even misguide clustering results. Moreover, reducing the number of features increases comprehensibility and ameliorates the problem of some unsupervised-learning algorithms failing with high-dimensional data.

Let's say we apply *k*-means with Euclidean distance as a measure for dissimilarity to cluster the data. For a finite amount of data, high dimensions lead to a sparse data space, and most of the data points will look equally far. For probability-based clustering algorithms, high dimensions mean more parameters to predict (that is, we need more data points to obtain accurate estimates). These clustering methods wouldn't work well in high dimensions.

To deal with high dimensionality, we can perform either feature transformation or feature selection. Principal component analysis, factor analysis, projection pursuit, and independent component analysis are examples of transformation methods, which involve transformations of the original variable space. In this article, I talk about selecting subsets of the original space. Subset selection is desirable in some domains that prefer the original variables so as to maintain these features' physical interpretation. In addition, feature transformation algorithms require computation or collection of all the features before dimension reduction can be achieved. In contrast, feature selection algorithms require computation or collection of only the selected feature subsets after the feature subsets are determined.

Carla Brodley and I define the goal of feature selection for unsupervised learning as finding the smallest feature subset that best uncovers "interesting natural" groupings (clusters) from data according to the chosen criterion.[1] Unlike supervised learning, which has class labels to guide the feature search, in unsupervised learning, we must define what interesting and natural mean in the form of criterion functions. The problem is that no global consensus exists on how to define interestingness. Moreover, different feature subspaces reveal different cluster structures. Which subspace should we pick?

Feature-selection approaches

Following supervised-learning terminology, you can categorize feature subset selection methods for unlabeled data as *filter* or *wrapper* approaches.

Filters

Filter methods use some intrinsic property of the data to select features without using the clustering algorithm that will ultimately be applied. The basic components are the feature search method and the feature selection criterion.

Filter methods must define feature relevance (interestingness) or redundancy without clustering the data. Manoranjan Dash and colleagues introduced a filter method that selects features on the basis of the entropy of distances between data points.[2] They observed that when the data are clustered, the distance entropy at that subspace should be low. Another filter method primarily for reducing redundancy is simply to cluster the features.

Wrappers

On the other hand, wrapper methods apply the unsupervised-learning algorithm to each candidate feature subset and then evaluate the feature subset by criterion functions that use the clustering result. Wrapper methods directly incorporate the clustering algorithm's bias in search and selection. The basic components are the feature search method, the clustering algorithm, and the feature selection criterion.

Brodley and I provide a survey of wrapper methods for unsupervised learning.[1] Most wrapper methods in that survey apply a feature-selection criterion similar to the one that the clustering algorithm optimizes. The clustering criterion deals with defining a similarity metric or defines what natural means. The feature-selection criterion defines interestingness. These two criteria need not be the same. Brodley and I examined two feature selection criteria—maximum likelihood and scatter separability—for a wrapper method that applies sequential forward search wrapped around Gaussian mixture model clustering.[1] To cluster data, we need to make assumptions and define natural grouping. With this model, we assume that each of our natural groups is Gaussian. Here, we investigate two ways to define interestingness: maximum likelihood criterion and scatter separability criterion. Maximum likelihood is the same criterion we used in our clustering algorithm. Maximum likelihood prefers the feature subspace that can be modeled best as a Gaussian mixture. We also explored scatter separability because you can use it with many clustering algorithms. Scatter separability is similar to the criterion function used in discriminant analysis. It measures how far apart the clusters are from each other normalized by their within-cluster distance. High values of maximum likelihood and scatter separability are desirable. We concluded that no one criterion is best for all applications.

We also investigated the issues involved in creating a general wrapper method where you can apply any feature search, clustering, and selection criteria.[1] We observed that using the same number of clusters throughout feature search isn't a good idea because different feature subspaces have different underlying numbers of natural clusters. So, the clustering algorithm should also incorporate finding the number of clusters in feature search. We also discovered that various selection criteria are biased with respect to dimensionality. We then introduced a cross-projection normalization scheme that any criterion function can use.

Using the same number of clusters throughout feature search isn't a good idea because different feature subspaces have different underlying numbers of natural clusters.

Other approaches

In addition, techniques exist that closely integrate feature selection within the clustering algorithm. Similar to wrapper approaches, they include the clustering algorithm within feature search. However, unlike traditional wrapper approaches that wrap feature selection around clustering, the feature search, clustering, and evaluation components are closely integrated into a single algorithm.

Subspace clustering. Rakesh Agrawal and his colleagues introduced CLIQUE (*Cl*ustering *i*n *Que*st), a subspace-clustering algorithm that proceeds level-by-level from one feature to the highest dimension or until it generates no more feature subspaces with clusters (regions with high density points).[3] The idea is that dense clusters in dimensionality d should remain dense in $d-1$. Subspace clustering also lets you discover different clusters from various subspaces and combine the results. Several new subspace clustering methods were developed after CLIQUE and summarized in a review by Lance Parson, Ehtesham Haque, and Huan Liu.[4]

Probabilistic model. Martin H. Law, Anil K. Jain, and Mario A.T. Figueiredo incorporate feature saliency as a missing variable in a finite-mixture model that assumes relevant features to be conditionally independent given the cluster component label and assumes irrelevant features to have a probability density identical for all components.[5] So, you can perform feature selection and clustering simultaneously in a single expectation-maximization iteration.

Coclustering. As we mentioned earlier, you can perform feature selection by clustering in the feature space to reduce redundancy. *Coclustering* has recently become popular because of research in microarray analysis. Coclustering is simply clustering the row (sample space) and column (feature space) simultaneously. Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha provide a review of the literature.[6]

Challenges

In feature selection for unlabeled data, we need to keep relevant features and remove redundancy. Different researchers have introduced varying criteria for feature selection. To define interestingness and relevance, researchers have proposed measures such as scatter separability, entropy, category utility, maximum likelihood, density, and consensus. The search process handles redundancy implicitly (for example, when adding new features, don't change the evaluation criterion) or explicitly through clustering in the feature space.

Defining interestingness is difficult because it's relative. Given the same data, what's interesting to a physician will differ from what's interesting to an insurance company. No single criterion is best for all applications. This led us to work in an interactive visualization environment where the user guides the feature search through visualization techniques and where feature selection serves as a visualization tool.[7] This has led researchers to explore ways to optimize multi-objective criteria. The difficulty of

defining interestingness has also prompted researchers to look at ensembles of clusters from different projections (or feature subspaces) and apply consensus of solutions to provide the final clustering.

Another direction is to look at feature selection with hierarchical clustering, which provides groupings at various perceptual levels. In addition, you might wish to develop algorithms that select a different feature subset for each cluster component. Coclustering and subspace clustering allow different subspaces for varying clusters. Another important direction is to capture structured relationships among features and clusters.

Evaluating clustering results remains a problem. The most common approach is to compare the discovered clusters to those of labeled classes through measures, such as mutual information and accuracy. However, this is just one interpretation of the data. The best solution is to work with domain experts, who can judge whether the discovered features and clusters make sense.

References

1. J.G. Dy and C.E. Brodley, "Feature Selection for Unsupervised Learning," *J. Machine Learning Research*, vol. 5, Aug. 2004, pp. 845–889.

2. M. Dash et al., "Feature Selection for Clustering—A Filter Solution," *Proc. 2002 IEEE Int'l Conf. Data Mining* (ICDM 2002), IEEE Press, 2002, pp. 115–122.

3. R. Agrawal et al., "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," *Proc. 1998 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, 1998, pp. 94–105.

4. L. Parson, E. Haque, and H. Liu, "Subspace Clustering for High Dimensional Data: A Review," *SIGKDD Explorations*, vol. 6, no. 1, 2004, pp. 90–105.

5. M.H. Law, M. Figueiredo, and A.K. Jain, "Feature Selection in Mixture-Based Clustering," *Advances in Neural Information Processing Systems 15*, MIT Press, 2003, pp. 609–616.

6. I.S. Dhillon, S. Mallela, and D.S. Modha, "Information-Theoretic Co-Clustering," *Proc. 9th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining* (KDD 03), ACM Press, 2003, pp. 89–98.

7. J.G. Dy and C.E. Brodley, "Interactive Visualization and Feature Selection for Unsupervised Data," *Proc. 6th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining* (KDD 00), ACM Press, 2000, pp. 360–364.

Variable Selection Using Ensemble Methods

Kari Torkkola, *Motorola*
Eugene Tuv, *Intel*

The emergence of extremely large data sets in various applications such as gene expression array analysis, proteomics, information retrieval, and text classification has recently made variable selection critical. Variable selection concerns modeling the relationship between a variable of interest Y and a subset of potential explanatory variables or predictors X. The problem is finding the best subset of predictors. Solving this can provide a better understanding of the underlying phenomena that generate the data. Typically, it also improves the predictive ability of a model inferred from the data.

Traditional multivariate statistics has approached variable selection using stepwise selection and best subset selection within linear-regression models. More recent trends are nonlinear models and addressing the question of instability (a small change in the data might result in a drastic change in the inferred results). Here we discuss an approach that addresses both of these concerns. We address nonlinearity using decision trees as the underlying regressors or classifiers, and instability by employing ensembles of decision trees.

Given a ranking of variables, it's not clear how to threshold the ranking to select only important variables and to exclude those that are mere noise. We present a principled approach to doing this using artificial-contrast variables.

Ensembles of trees for variable selection

You can divide variable-selection methods into three major categories. *Filter* methods evaluate some measure of relevance for all the variables and rank them on the basis of that measure (the measure might not be relevant to the task). Using some learner, *wrapper* methods actually learn the solution to the problem evaluating all possible variable combinations (this is usually computationally prohibitive for large variable sets). *Embedded* methods use a learner with all variables but infer the set of important variables from the learner's structure.

Decision trees are an example of nonlinear, fast, flexible base learners that can easily handle massive data sets of mixed-variable type. You can also consider a decision tree an embedded variable-selection mechanism because its node structure provides information about variables' importance. However, a single tree is inherently unstable. You can remedy this by using ensemble methods—for example, by bagging (combining the outputs of several base learners).[1]

Random Forest is a representative of tree ensembles that grows a forest of decision trees on bagged samples.[2] The randomness originates from sampling both the data and the variables. For each tree to be constructed, we draw a different sample of training data with replacement. The sample's size is the same as that of the original data set. This bootstrap sampling typically leaves 30 percent of the data out-of-bag. These data help provide an unbiased estimate of the tree's performance. Furthermore, we choose approximately $m = \sqrt{M}$ variables (M is the size of the variable set) randomly at the construction of each new split in the tree. The best split among these m variables is chosen for the current node, in contrast to typical decision tree construction, which selects the best split among all variables.

Adding up how often different variables were used in the splits of the tree (and from the quality of those splits) gives a measure of variable importance as a byproduct of the construction. For an ensemble of trees, we simply averaged the importance measure over the ensemble. The regularization effect of averaging makes this measure much more reliable.

Artificial contrasts

We can determine a cut-off point for a ranked list of variables on the basis of this assumption: A stable variable-ranking method, such as an ensemble of trees, that measures an input's relative relevance to a target variable Y would assign a significantly (in the statistical sense) higher rank to a legitimate variable X_i than to an artificial random variable created from the same distribution as X_i but independent of Y. We propose to obtain these artificial-contrast variables by randomly permuting values of original M variables across the N examples.

To increase the statistical significance, we compare all variables' importance scores to a percentile of importance scores of the N contrasts (we used the 75th percentile). A statistical test (student's t-test) compares the scores over T series. We select variables that score significantly higher than contrasts.

To allow detection of less important vari-

ables, we remove the effects of the newly discovered relevant variables on the Y. To accomplish this, a random forest predicts the target using only these relevant variables and computes a residual of the target. Then we repeat the process until no variables remain with scores significantly higher than the contrasts'. The last step is identical to stage-wise regression, but applied to a nonlinear model.

Figure 1 shows the *Artificial Contrasts with Ensembles* (ACE) algorithm for regression, which uses the following notation:

- Z: permuted versions of X,
- T: number of repeated permutations,
- F: current working set of variables,
- Φ: set of important variables,
- $\boldsymbol{V}_{i.}$: ith row of variable importance matrix $\boldsymbol{V}$,
- $\boldsymbol{V}_{.j}$: jth column of matrix $\boldsymbol{V}$,
- $g_I(F, Y)$: function that trains an ensemble of L trees based on variables F and target Y and returns a row vector of importance for each variable in F, and
- $g_Y(F, Y)$: function that trains an ensemble based on variables F and target Y and returns a prediction of Y.

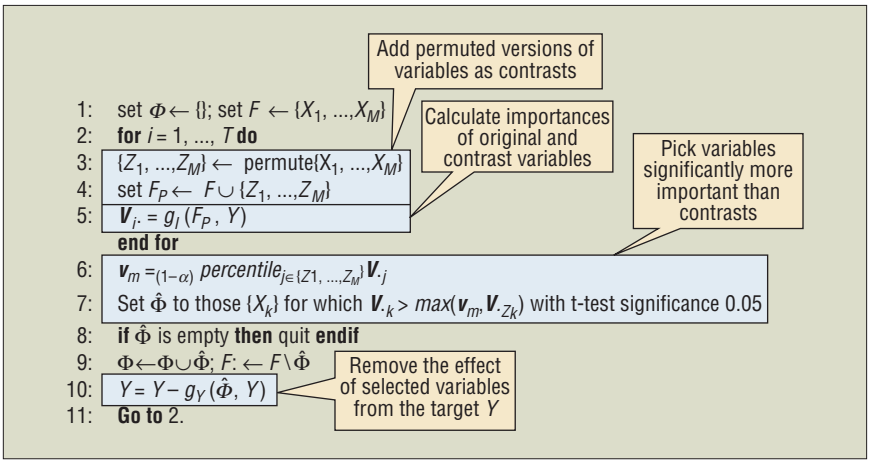


Figure 1. The Artificial Contrasts with Ensembles algorithm for regression.

Experiments

We tested how this method performs with two types of data (data sets having linear relationships and those having nonlinear relationships), both embedded in noise. Figure 2 displays illustrative benchmarking results. We compared five methods: ACE, RF (inherent variable ranking embedded in the Random Forest),[2] CFS (Correlation-Based Feature Selection),[3] CFS-Gen (the same with genetic search), and RFE (Recursive Feature Elimination).[4] Only ACE, CFS, and CFS-Gen automatically determine the number of important variables (ACE performs better than CFS!). RF and RFE produce a mere ranking, but we gave them the unfair advantage of cutting the ranking at the known number of important variables.

In the linear case, the target is a simple linear combination of a number of input variables plus noise $Y = -0.25x(1) + 0.1x(2) + 0.05x(3) + 0.025x(4) + 0.015x(5) + 0.01N(0, 1)$, where each $x(i)$ is drawn from $N(0, 1)$. Fifteen independent noise variables drawn from $N(0, 1)$ were joined to the data columns. In the nonlinear case, we used the generator that Jerome Friedman describes.[5] There are 10 important variables together with 10 independent noise variables. In both cases, the data set was 200 samples. The smaller the data set, the harder the variable selection problem becomes.

Conclusions

This approach to variable selection provides a truly autonomous method that considers all variable interactions and doesn't require a preset number of important variables. The method retains all the good features of ensembles of trees: it can use mixed-type data and tolerate missing variables, and it doesn't consider variables in isolation. The method is applicable to both classification and regression. It will report redundant variables if at least one of them is relevant

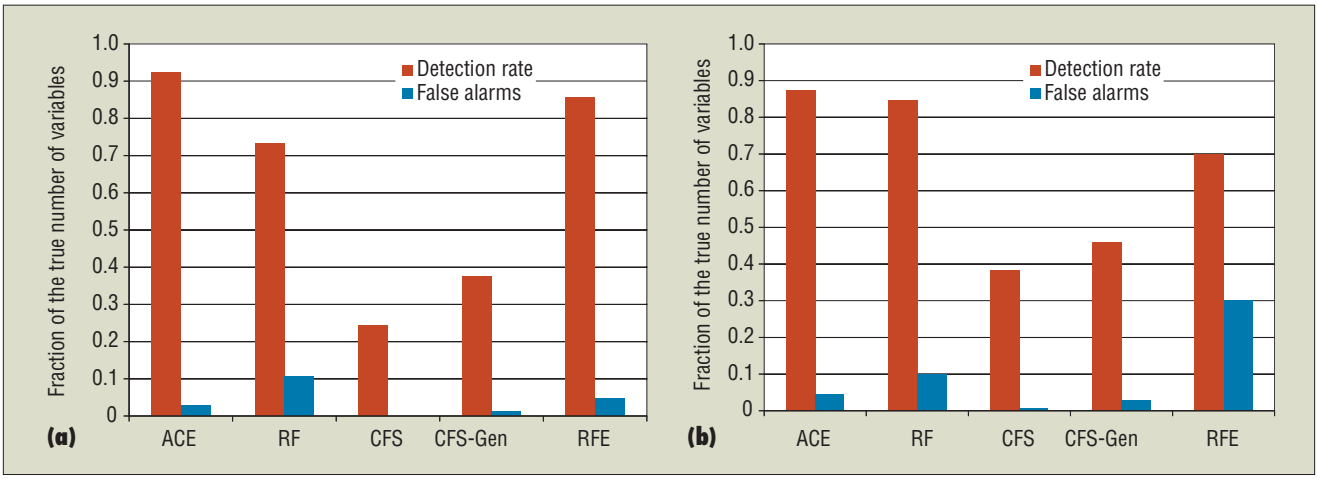


Figure 2. Results of applying the Artificial Contrasts with Ensembles (ACE) algorithm to data having (a) linear and (b) nonlinear relationships. The red bar denotes the method's detection rate—that is, of those variables that have a relationship with *Y*, the percentage that were detected as such. The blue bar displays the false alarm rate, the percentage of noise variables that were falsely detected as important. These numbers are averages over 50 data sets.

to a response. The computational complexity is that of Random Forest, $O(\sqrt{M}\ N \log N)$, where M is the number of variables and N is the number of observations. This is, in fact, lighter than that of any of the benchmark methods.

References

1. L. Breiman, "Bagging Predictors," *Machine Learning*, vol. 24, no. 2, 1996, pp. 123–140.
2. L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, 2001, pp. 5–32.
3. M. Hall, "Correlation-Based Feature Selection for Machine Learning," PhD thesis, Dept. of Computer Science, Waikato Univ., 1998.
4. I. Guyon et al., "Gene Selection for Cancer Classification using Support Vector Machines," *Machine Learning*, vol. 46, nos. 1–3, 2002, pp. 389–422.
5. J.H. Friedman, *Greedy Function Approximation: A Gradient Boosting Machine*, tech. report, Dept. of Statistics, Stanford Univ., 1999.

Minimum Redundancy–Maximum Relevance Feature Selection

Hanchuan Peng, Chris Ding, and Fuhui Long, *Lawrence Berkeley National Laboratory*

A critical issue in pattern analysis is feature selection. Instead of using all available variables (features or attributes) in the data, one selects a subset of features to use in the discriminant system. Feature selection has numerous advantages: dimension reduction to reduce the computational cost, noise reduction to improve classification accuracy, and more interpretable features or characteristics that can help, for example, to identify and monitor target diseases or function types. These advantages are important in applications such as gene marker selection for microarray gene expression profiles[1,2] and medical image morphometry.[3] For example, selecting a small set of marker genes could be useful in discriminating between cancerous and normal tissues.

Two general approaches to feature selection exist: *filters* and *wrappers*.[4] Filter methods select features on the basis of their relevance or discriminant powers with regard to the targeted classes. Simple methods based on mutual information and statistical tests (t-test, F-test) have proven effective. In this approach, feature selection isn't correlated to any specific prediction methods. So, the selected features have better generalization properties—that is, the selected features from training data generalize well to new data.

Wrapper methods wrap feature selection around a specific prediction method; the prediction method's estimated accuracy directly judges a feature's usefulness. One can often obtain a set with a very small number of features, which gives high accuracy because the features' characteristics match well with the learning method's. Wrapper methods typically require extensive computation to search the best features.

One common practice of filter methods is to simply select the top-ranked features—say, the top 50. A deficiency of this simple approach is that these features could be correlated among themselves. For the gene-marker selection problem, if gene g is ranked high for the classification task, the filter method will likely select other genes highly correlated with g. Simply combining one very effective gene with another doesn't necessarily form a better feature set, because the feature set contains a certain redundancy. Several recent studies have addressed such redundancy.[3,5,6]

This leads to minimum redundancy–maximum relevance (mRMR) feature selection;[1,2] that is, selected features should be both minimally redundant among themselves and maximally relevant to the target classes. The emphasis is direct, explicit minimization of redundancy.

mRMR feature selection

For categorical features (variables), we use mutual information to measure the level of similarity between features. Let S denote the features subset that we're seeking and Ω the pool of all candidate features. The minimum redundancy condition is

$$\min_{S\subset\Omega} \frac{1}{|S|^2} \sum_{i,j\in S} I(f_i, f_j) \tag{1}$$

where $I(f_i,f_j)$ is mutual information between f_i and f_j, and $|S|$ is the number of features in S.

To measure features' level of discriminant power when they're differentially expressed for different targeted classes, we again use mutual information $I(c, f_i)$ between the targeted classes $c = \{c_1, \ldots, c_K\}$ (we call c the classification variable) and the feature f_i. So, $I(c, f_i)$ quantifies the relevance of f_i for the classification task. The maximum relevance condition is to maximize the total relevance of all genes in S:

$$\max_{S\subset\Omega} \frac{1}{|S|} \sum_{i\in S} I(c, f_i) \tag{2}$$

We obtain the mRMR feature set by optimizing these two conditions simultaneously, either in quotient form

$$\max_{S\in\Omega} \left\{ \sum_i I(c, f_i) \Big/ \left[\frac{1}{|S|} \sum_{i,j\in S} I(f_i, f_j) \right] \right\} \tag{3}$$

or in difference form

$$\max_{S\in\Omega} \left\{ \sum_{i\subset S} I(c, f_i) - \left[\frac{1}{|S|} \sum_{i,j\in S} I(f_i, f_j) \right] \right\} \tag{4}$$

The exact solution to mRMR requires $O(N^{|S|})$ search to obtain (N is the number of features in Ω). In practice, a near-optimal solution is sufficient, which the incremental-search algorithm obtains. The first feature is selected according to equation 3 or 4—that is, the feature with the highest $I(c, f_i)$. The rest of the features are selected incrementally. The solution can be computed efficiently in $O(|S|\cdot N)$.

For features taking continuous values, we compute quantities such as the F-statistic between features and the classification variable c as the score of maximum relevance

$$\max_{S\subset\Omega} \frac{1}{|S|} \sum_{i\in S} F(f_i, c) \tag{5}$$

and the average Pearson correlation coefficient of features as the score for minimum redundancy,

$$\min_{S\subset\Omega} \frac{1}{|S|^2} \sum_{i,j} \left| c(f_i, f_j) \right| \tag{6}$$

where we assume that both high positive and high negative correlation mean redundancy. We can also consider the distance function $d(f_i, f_j)$ (for example, L_1 distance) for the minimum redundancy condition:

$$\max_{S\subset\Omega} \frac{1}{|S|^2} \sum_{i,j\in S} d(f_i, f_j) \tag{7}$$

Mutual information formalism

As a theoretical basis of mRMR feature selection, we consider a more general feature-selection criterion, *maximum dependency* (MaxDep).[2] In this case, we select the feature set $S_m = \{f_1, f_2, \ldots, f_m\}$, of which the joint statistical distribution is maximally dependent on the distribution of the classification variable c. A convenient way to measure this statistical dependency is mutual information,

$$I(S_m; c) = \iint p(S_m, c) \log \frac{p(S_m, c)}{p(S_m)\,p(c)} dS_m dc \quad (8)$$

where $p(.)$ is the probabilistic density function. The MaxDep criterion aims to select features S_m to maximize equation 8. Unfortunately, the multivariate density $p(f_1, \ldots, f_m)$ and $p(f_1, \ldots, f_m, c)$ are difficult to estimate accurately when the number of samples is limited, the usual circumstance for many feature selection problems. However, using the standard multivariate mutual information

$$J(y_1, \cdots y_n) = \iint p(y_1, \cdots y_n) \log \frac{p(y_1, \cdots, y_n)}{p(y_1) \cdots p(y_m)} \boldsymbol{d}y_1 \cdots dy_n \quad (9)$$

we can factorize equation 8 as

$$I(S_m; c) = J(S_m, c) - J(S_m). \quad (10)$$

Equation 10 is similar to the mRMR feature selection criterion of equation 4: The second term requires that features S_m are maximally independent of each other (that is, least redundant), while the first term requires every feature to be maximally dependent on c. In other words, the two key parts of mRMR feature selection are contained in MaxDep feature selection.

Experiments on gene expression data

We've found that explicitly minimizing the redundancy term leads to dramatically better classification accuracy. For example, for the lymphoma data in figure 3a, the commonly used MaxRel features lead to 13 *leave-one-out cross-validation* errors (about 86 percent accuracy) in the best case. Selecting more than 30 mRMR features results in only one LOOCV error (or 99.0 percent accuracy). For the lung cancer data in figure 3b, mRMR features lead to approximately five LOOCV errors, while maxRel features lead to approximately 10 errors when more than 30 features are selected. We present more extension results elsewhere.[1,2] The performance of mRMR features is good, especially considering that the features are selected independently of any prediction methods.

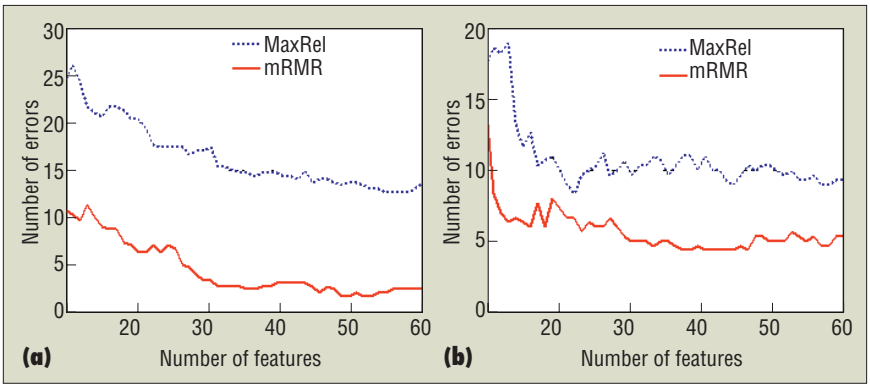


Figure 3. Average leave-one-out cross-validation errors of three different classifiers—Naïve Bayes, Support Vector Machine, and Linear Discriminant Analysis—on two multiclass data sets, lymphoma (a) and lung cancer (b), which contain microarray gene expression profiles. Lymphoma: 4,026 genes and 96 samples for nine subtypes of lymphoma; Lung cancer: 918 genes and 73 samples for seven lung cancer subtypes. More information on these data sets is available elsewhere.[1,2]

Extension

The mRMR feature-selection method is independent of class-prediction methods. One can combine it with a particular prediction method.[2] Because mRMR features offer broad coverage of the characteristic feature space, one can first use mRMR to narrow down the search space and then apply the more expensive wrapper feature-selection method at a significantly lower cost.

Acknowledgments

Chris Ding's work is partially supported by the Office of Science, US Department of Energy, under contract DE-AC03-76SF00098.

References

1. C. Ding and H.C. Peng, "Minimum Redundancy Feature Selection from Microarray Gene Expression Data," *Proc. IEEE Computer Soc. Bioinformatics Conf.* (CSB 03), IEEE CS Press, 2003, pp. 523–528.

2. H.C. Peng, F.H. Long, and C. Ding, "Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, 2005, pp. 1226–1238.

3. E. Herskovits, H.C. Peng, and C. Davatzikos, "A Bayesian Morphometry Algorithm," *IEEE Trans. Medical Imaging*, vol. 23, no. 6, 2004, pp. 723–737.

4. R. Kohavi and G. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence*, vol. 97, nos. 1–2, 1997, pp. 273–324.

5. J. Jaeger, R. Sengupta, and W.L. Ruzzo, "Improved Gene Selection for Classification of Microarrays," *Proc. 8th Pacific Symp. Biocomputing* (PSB 03), World Scientific, 2003, pp. 53–64.

6. L. Yu and H. Liu, "Efficiently Handling Feature Redundancy in High-Dimensional Data," *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining* (KDD 03), ACM Press, 2003, pp. 685–690.

Fostering Biological Relevance in Feature Selection for Microarray Data

Michael Berens, *Translational Genomics Research Institute*
Huan Liu, Lance Parsons, and Zheng Zhao, *Arizona State University*
Lei Yu, *State University of New York, Binghamton*

Microarray-based analysis techniques that query thousands of genes in a single experiment present unprecedented opportunities and challenges for data mining.[1] Gene filtering is a necessary step that removes noisy measurements and focuses further analysis on gene sets that show a strong relationship to phenotypes of interest. The problem becomes particularly challenging because of

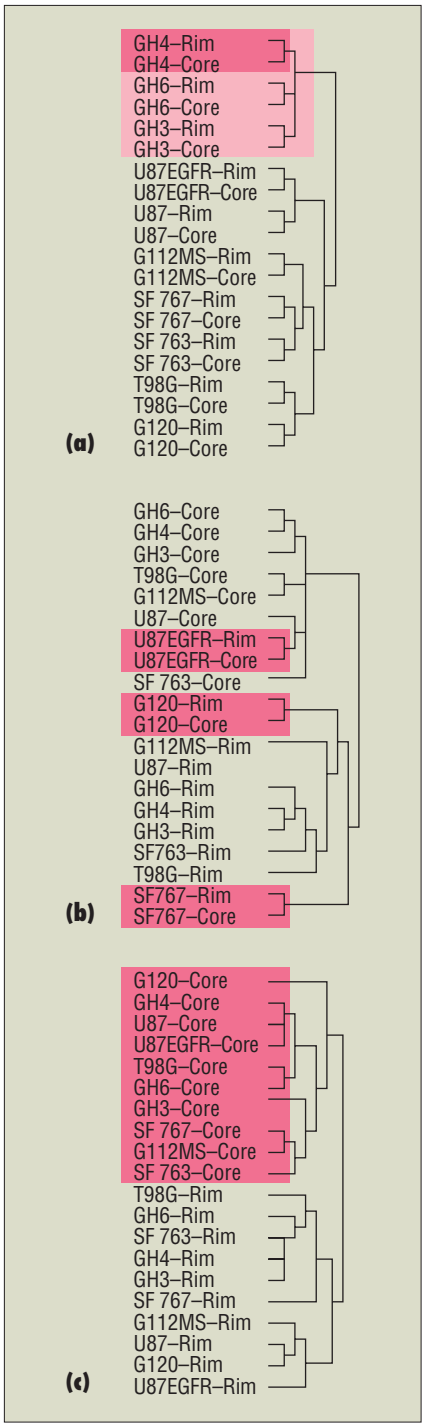


Figure 4. Hierarchical clustering of core and rim samples from 10 glioma cell lines. (a) clustering using all genes (features), (b) clustering using 22 genes selected by 2-sample t-test, and (c) after supervised feature selection, core and rim samples are clustered together, respectively.

the large number of features (approximately 30,000–40,000 genes) and the small number of samples (about 100 experiments). So, dimensionality reduction is necessary to enable effective data mining such as classification, clustering, or discriminant analysis. Feature selection, a technique that selects a subset of features from the original ones, is a frequently used preprocessing technique in data mining.[2,3]

A recent experiment on glioma cell line data reveals the importance of feature selection in microarray analysis.[4] By applying hierarchical clustering, we can visualize the discriminative power of various gene sets emerging from the two phenotypes' gene expression profiles. Figure 4a shows the dendrogram generated by hierarchical clustering based on all of the genes. Core and rim samples from the same specimen are uniformly grouped together, indicating that the core-to-rim variations are less significant than specimen-to-specimen variations. The two-sample t-test is commonly used to identify genes showing differential expression and selects 22 genes with p-values < 0.01. Figure 4b shows the dendrogram produced using these 22 genes; the clusters in red boxes still contain both core and rim samples. After the application of supervised feature selection,[5] the core-to-rim variations are far more pronounced and the samples cluster neatly into a core cluster and a rim cluster (see figure 4c). The clustering results indicate that feature selection selects discriminatory features better than statistical criteria such as a t-test do.

Beyond statistical significance

Machine learning and statistical approaches can effectively identify both statistically relevant genes and those with redundant information. However, many statistically significant patterns found in data sets with a huge feature space and few samples might not be biologically relevant. Microarray studies' goal is often to determine which genes and pathways determine a target phenotype or clinical condition. In other words, statistically significant patterns are interesting, but it would be even better if these patterns could help identify genes with biological relevance.

A high-level goal of microarray analysis is to elucidate the developmental model of the phenotypes under study. Researchers use microarray experiments to identify genes and pathways for further study (for example, to find potential drug targets). Researchers might wish to develop diagnostic or prognostic tools, which are practical only when the number of genes is small and the classification is robust across many samples and noise levels. Suitable genes and pathways are those with not only statistical significance in the data but also certain biological or molecular traits. The additional downstream requirements necessitate the evaluation of not only microarray data but also factors such as the availability of antibodies for a given protein or the ability to interrupt a pathway with minimal harmful side effects.

The complexities of biological information can often mean that the class labels might be unreliable or too coarse, suggesting the use of unsupervised or semisupervised techniques. For example, a class label might be the histological categorization of a cancer. While those categories are quite useful, they often don't tell the entire story. Histologically similar cancers can, in fact, be molecularly distinct, with different underlying causes and clinical outcomes.

Fostering biological relevance

We define three types of biological relevance:

- genes with known functions, which contribute to learning efficiency,
- genes with unknown functions, which present opportunities to contribute to high-impact results, and
- genes that are known to be good targets a priori (for example, genes with readily available antibodies or those suspected owing to independent evidence).

We developed a tool, Reporter-Surrogate Variable Program, which reduces the number of selected genes while increasing the overall discriminative power and helps biologists select more biologically relevant genes for subsequent biological and clinical validation.[4] Specifically, RSVP identifies a small subset of reporter genes that are mutually nonredundant and jointly provide a profile for discriminating the two phenotypes under study. In addition, for each reporter gene, RSVP identifies and presents a set of surrogate genes that are highly correlated to the reporter gene. So, biologists can replace reporter genes with genes from the surrogate lists that provide greater biological relevance without jeopardizing the overall discriminative power. RSVP aims to produce results that are both statistically

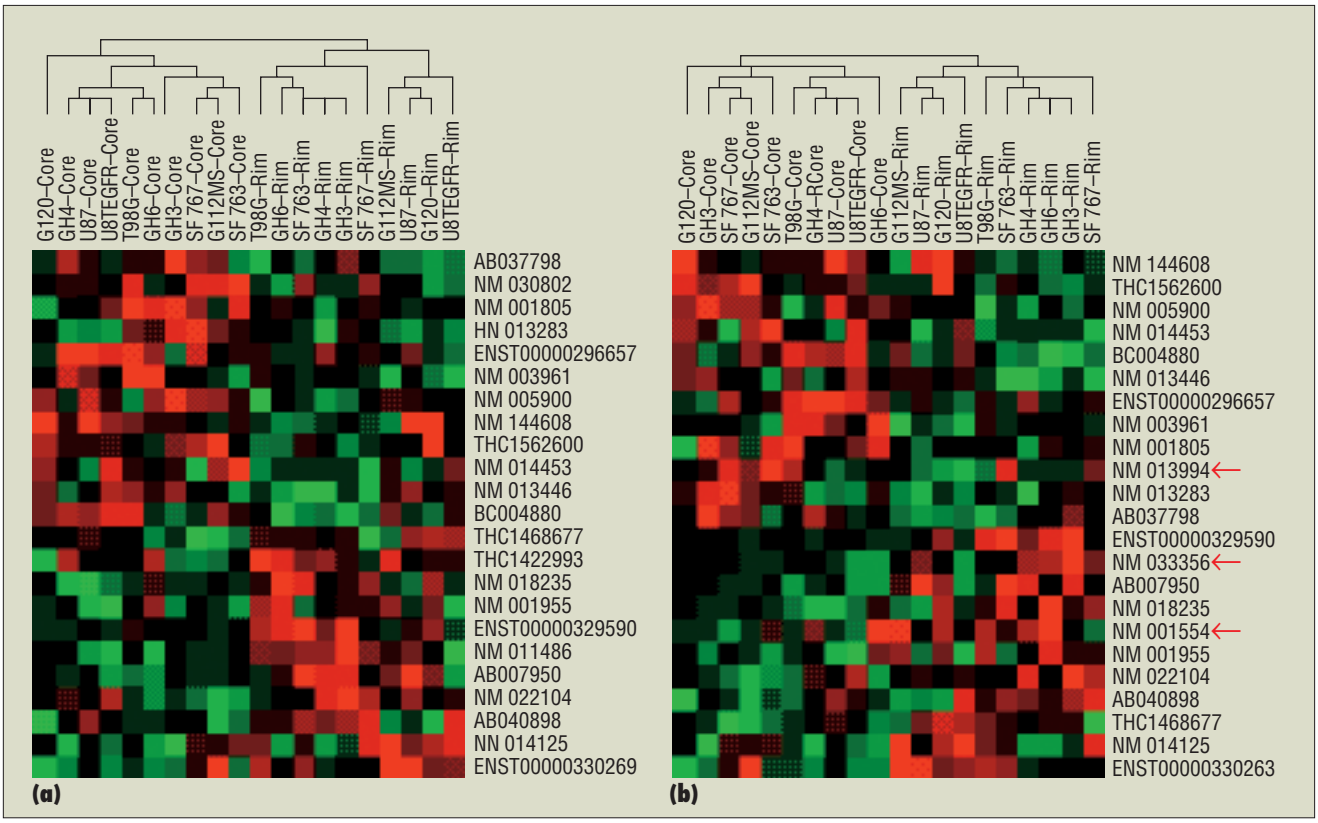


Figure 5. Hierarchical clustering results based on genes selected by the Reporter-Surrogate Variable Program tool: (a) a dendogram with an expression heatmap from the 23 reporter genes, and (b) a similar result from 20 reporter genes and three surrogate genes of biological relevance, replacing three reporter genes.

significant and biologically relevant.

RSVP identified 23 reporter genes and their corresponding surrogate genes from the 306 genes selected by the two-sample t-test ($p < 0.1$). Figure 5a shows the clustering dendrogram and a heatmap based on the 23 reporter genes' expression values. In the heatmap, log ratios of 0 are black, and increasingly positive or negative log ratios are increasingly red or green, respectively. The 20-sample dendrogram forms two distinct clusters corresponding to the two phenotypes. Simply removing the reporter genes with unknown functions or replacing them with randomly selected genes resulted in reduced discriminative power. However, simultaneously replacing the three unknown reporter genes (NM_014486, THC1422993, NM_030802, marked by arrows) with their surrogate genes with known functions produced very similar cluster results, as figure 5b shows. Coexpression of genes in the reporter gene set and surrogate lists might also help reveal the functions of many genes for which such information is currently unavailable.

Feature selection with clinical impact

Enriching statistically significant gene lists with biologically relevant genes can help expedite biological discovery and downstream analysis. Despite public knowledge bases' increasing accessibility, the process remains largely manual, with little consistency among researchers or labs. By incorporating additional biological knowledge directly into feature selection, we can automate much of the process and improve researchers' ability to leverage the increasing amounts of publicly available research data. Interdisciplinary researchers could limit results to those targets for which antibodies are readily available to enable further study. Researchers can also more easily target drug research to particular locations in the cell. As microarray techniques advance into DNA and protein research, the number of features is increasing to millions. Sophisticated feature selection techniques that can leverage existing domain knowledge will become even more important.

References

1. G. Piatetsky-Shapiro and P. Tamayo, "Microarray Data Mining: Facing the Challenges," *SIGKDD Explorations Newsletter*, vol. 5, no. 2, 2003, pp. 1–5.

2. H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering," *IEEE Trans. Knowledge and Data Eng.*, vol. 17, no. 3, 2005, pp. 1–12.

3. J.L. Rennert et al., "Supervised Pattern Recognition Identifies Principal Components of Differential Gene Expression Related to Brain Tumor Migration," *Proc. Oncogenomics 2005: Dissecting Cancer Through Genome Research,* Am. Assoc. for Cancer Research, 2005, p. B38.

4. L. Yu et al., *Exploiting Statistical Redundancy in Expression Microarray Data to Foster Biological Relevancy*, tech. report TR-05-005, Computer Science and Eng. Dept., Arizona State Univ., 2005.

5. L. Yu and H. Liu, "Redundancy Based Feature Selection for Microarray Data," *Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining* (KDD 04), 2004, pp. 737–742.

Feature Selection: We've Barely Scratched the Surface

George Forman, *Hewlett-Packard Labs*

Selecting which inputs to feed into a learning algorithm is important but often underappreciated. People usually talk about "the" clusters in a data set as if there were one set of them. But if you were to cluster, for example, the vehicles in a parking lot into groups, your answer would depend completely on the features you considered: color? model? license plate? Without prior knowledge of which sorts of clusters are desired, no right or wrong choice exists. However, if someone paid you to generate a predictive model for gas mileage, you would consider vehicle weight and ignore color. These examples are meant to be obvious, but real-world data sets tend to involve large and often complex feature selection choices, whether or not they're made deliberately.

If feature selection is done poorly, no clever learning algorithm can compensate—for example, predicting gas mileage from color and trim. If done well, the computational and memory demands of both the inducer and the predictor can be reduced, and usually more important, the prediction accuracy improved. The performance of naïve Bayes—ever popular for its ease of programming—is highly sensitive to feature selection; even relatively insensitive algorithms, such as support vector machines, can benefit substantially (see the sidebar). In some circumstances, such as biochemistry wet labs, eliminating all but the essential features can reduce the cost of obtaining measurements. Finally, feature selection by itself has useful applications, such as the statistically improbable phrases now appearing at www.amazon.com to help end-users characterize books.

While several good feature-selection techniques exist, I contend that feature selection is still in its infancy and major opportunities await. (For a survey on feature selection, refer to the 2003 special issue on variable and feature selection in the online *Journal of Machine Learning Research* (http://jmlr.csail.mit.edu/papers/special/feature03.html) or to the recent survey by Huan Liu and Lei Yu.[1])

Low-hanging fruit

A first avenue is simply to bring known successful techniques into mainstream use. Too often an available data set is used as-is with all its features, no matter how they came to be. People generally give much more thought to the induction algorithms than to the features. Part of the solution lies in just streamlining user interfaces to make automated feature selection part of the natural process.

Of course, people don't want to be bothered with more knobs to tune. Just as you can use cross-validation to select which of several learning models performs best for a given training set, so too can it automate decisions about feature selection. (Cross-validation involves breaking a data set into, say, 10 pieces, and on each piece testing the performance of a predictor trained from the remaining 90 percent of the data. In this way, you can estimate how well each of several learning algorithms performs on the available data and then choose the best method to apply to all of the training data.) But this has its limits. Cross-validation on large data sets can exceed the user's patience budget, and cross-validation on small training sets is more likely to produce overfit models than true improvements in generalization accuracy. You can combat this with knowledge about which combinations of feature selection and learning algorithms perform well for different kinds of data. This is an open opportunity for metalearning research.

Several trends will increase the demand for feature selection. One is obviously the growing size of data sets, requiring either random subsampling of rows or purposeful feature selection of columns.

Accuracy vs. robustness

While a great deal of machine learning research seeks to improve accuracy, it sometimes comes at a cost in brittleness. To enable more widespread use of feature selection, there's a valuable vein of research in developing robust techniques. We at Hewlett-Packard have faced industrial data sets where most feature selection techniques fail spectac-ularly. For example, in a multiclass task for document classification where one class is very easy to predict—for example, German documents—most feature-selection methods will focus on the many strongly predictive foreign-word features for the easy class, leaving the other classes hard to distinguish.[2] Although we devised a solution for this specific type of problem, certainly more research into robust methods is necessary. I urge practitioners to share the failures they encounter on real data sets; most public benchmark data sets don't expose these issues.

Trends

I predict several trends will increase the demand for feature selection. One is obviously the growing size of data sets, requiring either random subsampling of rows or purposeful feature selection of columns. The former is easier, but the latter may be more beneficial. Feature selection might be the only reasonable choice for reducing wide data sets with many more columns than rows (for example, often more than 100,000 features in genomics or document classification).

And data sets are generally widening, with the increasing ability to link to additional databases and join with other tables. In my car example, you could link each vehicle to external databases with pollution ratings, sales figures, and review articles, potentially adding thousands of features. Today such linking requires human thought and effort, but tomorrow it could be automated.[3] This increases the pressure on automated feature selection to efficiently determine which widening is useful. The demand for this research will come primarily from practitioners who seek optimal prediction for economically valuable tasks, not from pure machine-learning researchers who care about optimizing performance on fixed, self-contained benchmark data sets for comparable, publishable results.

Rich data types

The trend toward richer data types is pushing feature selection in both scale and complexity. Natural language text features and image features are becoming commonplace—for example, in the relatively mature area of document classification. To handle rich data types, a feature generator replaces them with many features of primitive data

Feature Selection Improves Text Classification Accuracy

In the field of text classification, individual words serve as input features to machine learning classifiers. By selecting a good subset of words, we can improve the predictions. In the figure below, I illustrate this benefit for two popular learning algorithms: Naïve Bayes and Support Vector Machines.

The vertical axis shows the F-measure—a kind of accuracy measurement for rare classes—averaged over a large benchmark data set. The horizontal axis varies the number of top-ranked features (words) that are given to the learning model, even to the point of including all features (no feature selection). By selecting the optimal number of top-ranked features, the performance can be substantially improved over using all the many thousands of available word features, especially for Naïve Bayes. Methods for ranking features are many, and some can perform substantially worse, depending on the data set.

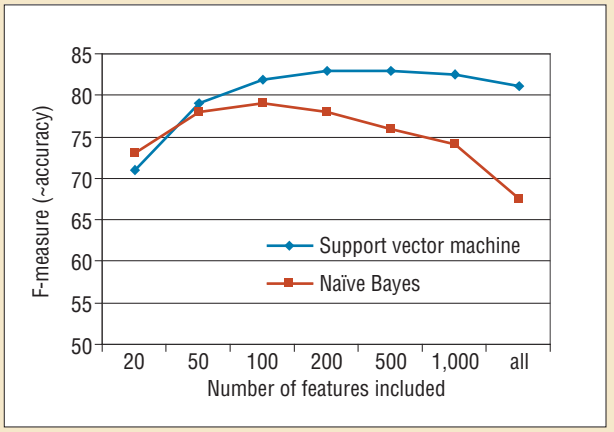


Figure A. Classification accuracy as a function of the number of words selected.[1]

Reference

1. G. Forman, "A Pitfall and Solution in Multi-Class Feature Selection for Text Classification," *Proc. 21st Int'l Conf. Machine Learning* (ICML 04), ACM Press, 2004, p. 38.

types. In the bag-of-words model for text, each unique word in the collective corpus generates a unique feature—for example, an integer representing the number of times that word occurred in each record. The number of generated features can become quite large if the vocabulary in the corpus is large, especially when multiple natural languages are present. (The widespread text-processing techniques of lowercasing, eliminating common stop words, and stemming words to their root form reduce the total number of features by only a small amount.) Or there might be multiple text fields to be expanded into separate sets of features, since the word "Smith" appearing in the title should be treated differently than if it appears in the author field.

Considering the rich, expressive power of human language to address any topic, a simple bag-of-words model gives a limited view—the words' relative positions are lost (Try reading with the words sorted: reading sorted the Try with words). By adding a feature for each two- and three-word phrase that appears in the corpus, the bag-of-phrases representation can distinguish a "light car" from a "car light" at the cost of many more potential features to consider. Other feature-generation techniques might link words and phrases to external databases with additional information to generate even more features, such as thesauri and controlled-vocabulary taxonomies. With the deluge of hierarchically nested XML data types and time-based multimedia, feature-generation research will continue to expand the possibilities.

In all, the potential space for feature generation from rich data types is enormous and not all worthwhile. Rather than attack it simply in terms of greater scale, there will be a need to integrate feature selection with feature generation, just as conventional breadth-first and A* search techniques carefully coordinate state generation with evaluation. After all, inducing predictive models can be stated as a search problem that considers variations in feature generation, feature selection, induction algorithms, and their associated parameters. While this might sound quite involved, CPU cycles will increasingly be cheaper than an expert's time.

That said, we can quickly fall into the trap of overfitting our data if our search space is large and the training set relatively small. For example, given only a few training examples, it might happen that color can help predict gas mileage in cross-validation, but we wouldn't expect this correlation to generalize to larger data sets. Once again, metalearning methods are called for to help guide the search in the absence of large amounts of training data for a particular new problem. Likewise, machine-readable domain knowledge could help constrain the search to meaningful correlations. We don't know how to automate this today, but hopefully we will one day.

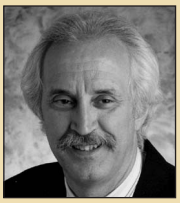
Cost and time

As if the present challenges weren't enough, real-world problems not uncommonly include a cost and time delay for obtaining (additional) features. For example, Sriharsha Veeramachaneni and his colleagues describe a practical biomedical problem where additional medical tests might provide predictive features at a cost.[4] They go on to develop an elegant algorithm to maximize predictive performance cost-efficiently. This is in contrast to typical active learning problems where the cost is entirely in obtaining class labels.

Time plays an additional role in some nonstationary domains where the best features have a seasonal dependency. For example, in spam filtering, the word "Christmas" is useful in December but has a fairly low value for the following months. Many similar time-related issues are associated with click stream mining of Web sites and shopping behavior.

I can't claim that cost and time represent current trends in research, but I foresee their need in practical deployment and expect that these areas will eventually see greater activity.

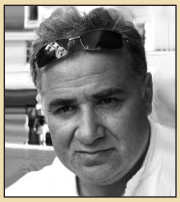
Huan Liu is an associate professor of computer science and engineering at Arizona State University. Contact him at hliu@asu.edu.



Edward R. Dougherty is a professor in the Department of Electrical Engineering and director of the Genomic Signal Processing Laboratory at Texas A&M University and director of the Computational Biology Division of the Translational Genomics Resesarch Insititute in Phoenix. Contact him at edward@ee.tamu.edu.

Jennifer G. Dy is an assistant professor in the Department of Electrical and Computer Engineering at Northeastern University. Contact her at jdy@ece.neu.edu; www.ece.neu.edu/faculty/jdy.

Kari Torkkola is a Distinguished Member of the technical staff at Motorola Labs. Contact him at Kari.Torkkola@motorola.com.

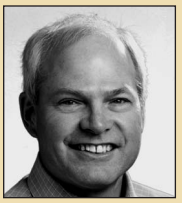


Eugene Tuv is a senior staff research scientist at Intel. Contact him at eugene.tuv@intel.com.

Hanchuan Peng is a research scientist in the Genomics Division, Lawrence Berkeley National Laboratory. Contact him at hpeng@lbl.gov.

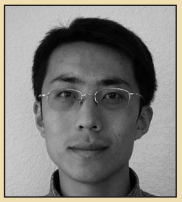
Chris Ding is a staff computer scientist at the Lawrence Berkeley National Laboratory. Contact him at chqding@lbl.gov.

Fuhui Long is a research scientist in the Life Science Division, Lawrence Berkeley National Laboratory. Contact her at flong@lbl.gov.

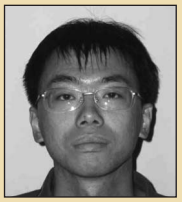


Michael Berens is a senior investigator in the Neurogenomics Division of the Translational Genomics Research Institute. Contact him at mberens@tgen.org.

Lance Parsons is a graduate student in the Department of Computer Science and Engineering in the Fulton School of Engineering at Arizona State University. Contact him at lparsons@asu.edu.



Lei Yu is an assistant professor in the Department of Computer Science at State University of New York at Binghamton. Contact him at lyu@cs.birmingham.edu.



Zheng Zhao is a PhD candidate in the Department of Computer Science and Engineering at Arizona State University. Contact him at zheng.zhao.1@asu.edu.

George Forman is a senior research scientist at Hewlett-Packard Labs. Contact him at ghforman@hpl.hp.com.

A vision

One reason C4.5 decision trees are popular is that they can handle a heterogeneous collection of feature types (mixed nominal, integer, and real) without requiring special user consideration. Although I stated earlier that people often pay too little attention to feature selection, no special user consideration will be necessary in my vision of future machine learning platforms. Instead, a robust feature selection subsystem equipped with metaknowledge will seamlessly handle heterogeneous types, linked-database widening, and so on. Getting there will require much stimulating research, fueled by real-world problems brought to light by practitioners. Any takers? ◼

References

1. H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering," *IEEE Trans. Knowledge and Data Eng*., vol. 17, no. 4, 2005, pp. 491–502.
2. G. Forman, "A Pitfall and Solution in Multi-Class Feature Selection for Text Classification," *Proc. 21st Int'l Conf. Machine Learning* (ICML 04), ACM Press, 2004, p. 38.
3. X. Yin and J. Han, "Efficient Classification from Multiple Heterogeneous Databases," *Proc. 9th European Conf. Principles and Practices of Knowledge Discovery in Databases* (PKDD 05), 2005, pp. 404–416.
4. S. Veeramachaneni et al., "Active Sampling for Knowledge Discovery from Biomedical Data," *Proc. 9th European Conf. Principles and Practice of Knowledge Discovery in Databases* (PKDD 05), 2005, pp. 343–354.